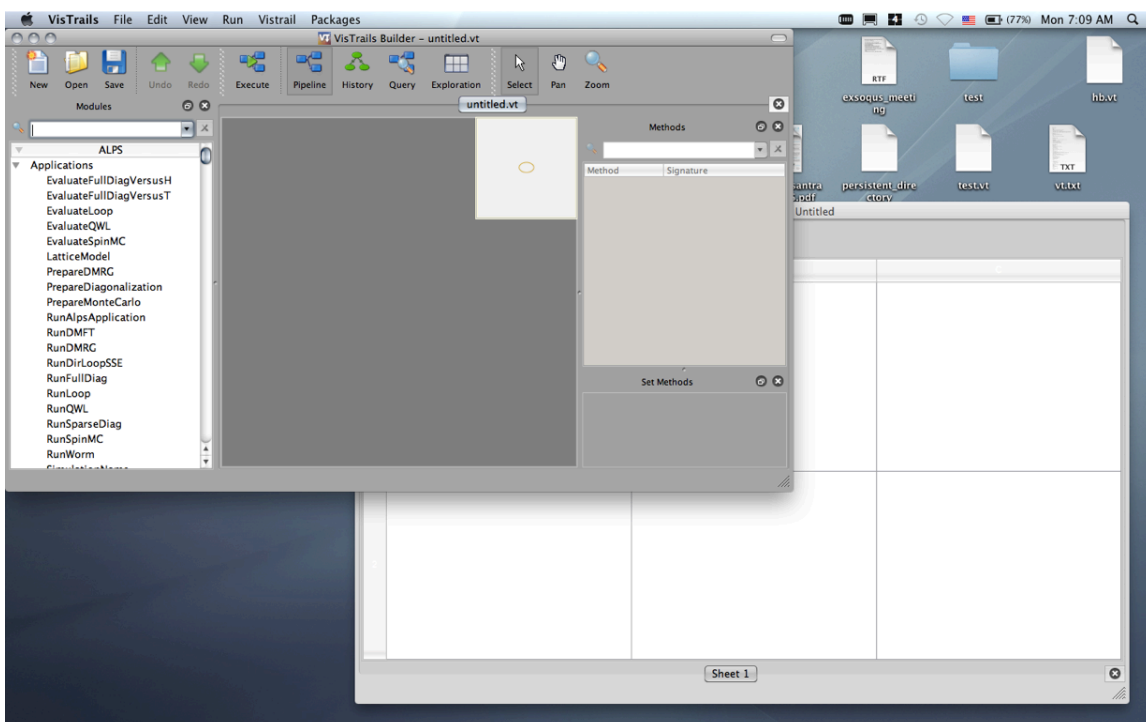


# VisTrails Introduction

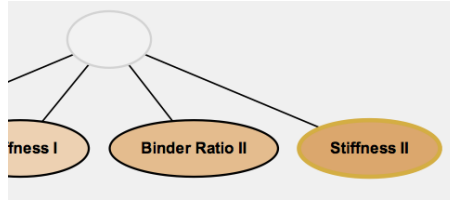
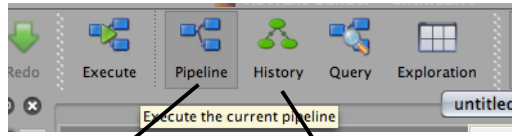
Bela Bauer ([bauerb@phys.ethz.ch](mailto:bauerb@phys.ethz.ch))



## Vistrails windows



# Views



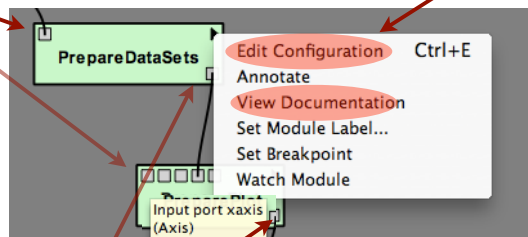
# Modules

Important for source modules!

Input ports

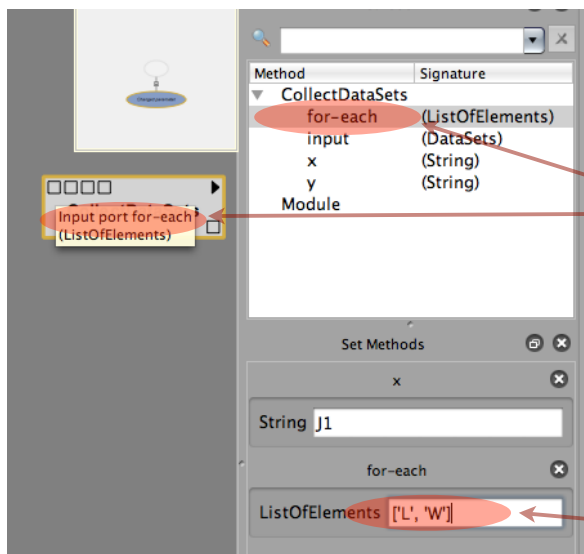
Ports:

- Input/Output
- Name
- Type:
  - VT module class
  - Connects only matching types
  - Weak typing



Output ports

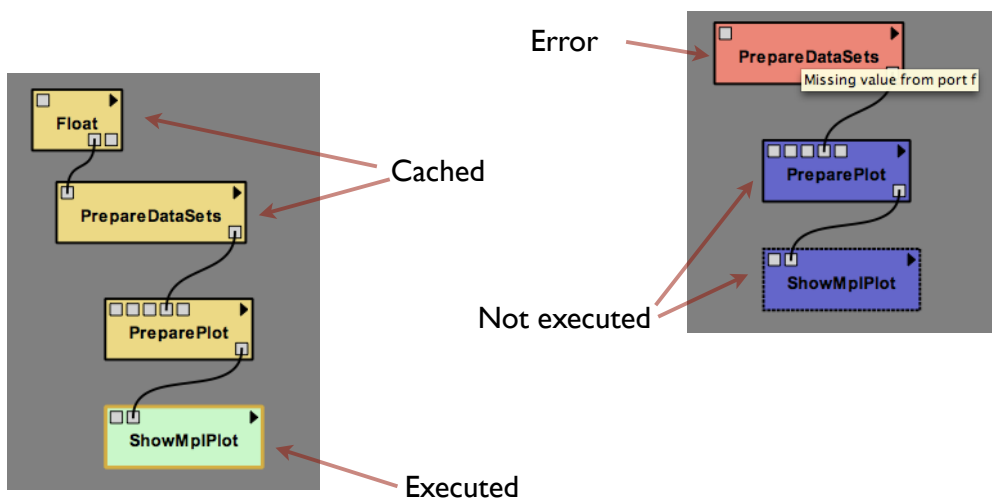
# More on modules



Input ports can often be set from other modules or directly

Most types are entered in the syntax of the Python interpreter!

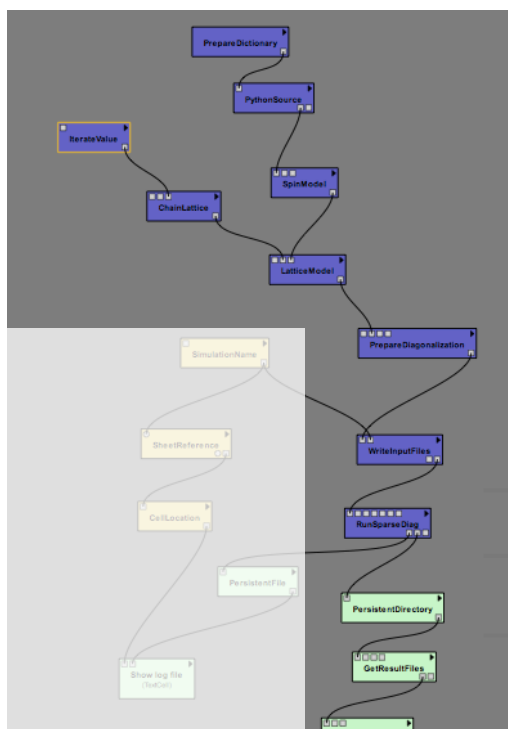
# Pipeline execution



# Caching

- Memory caching
  - Fully automatic
  - Lost after restarting VisTrails
- Disk caching (*persistent cache*)
  - Needs to be enabled by user
  - Remains “forever”
  - `PersistentFile/PersistentDirectory` modules

# Persistence



Input preparation

Running the simulation

Storing to persistence

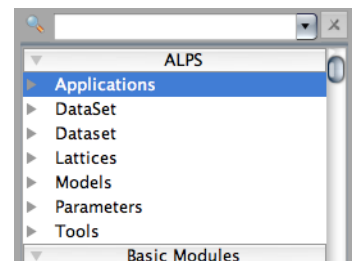
Data analysis

# More persistence

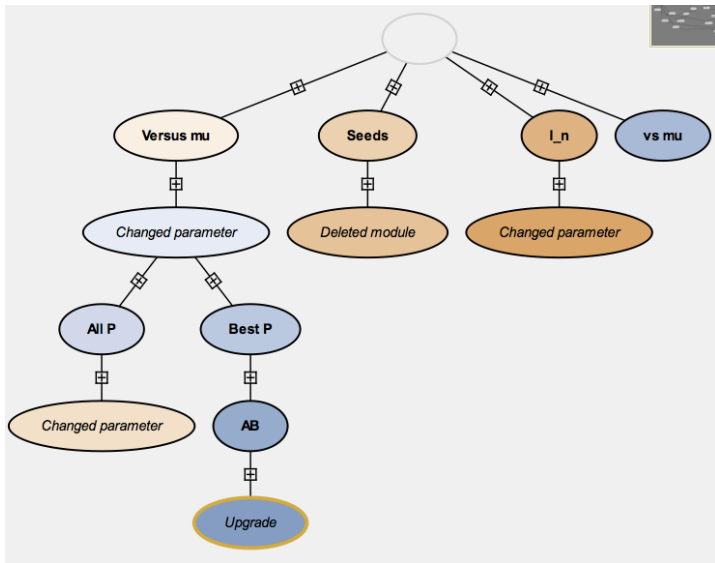
- Where does it go?
  - `~/vistrails/persistent_files`
- Other locations can be set in Preferences

# ALPS modules

- Library/applications-related modules
  - Gives access to ALPS applications with preparation of models, lattices, etc.
  - Closely related: Parameters modules
- DataSet modules
  - Independent of ALPS library and applications
  - 'Scriptable plotting and data analysis tool'



# History



- Instead of separate files, you can maintain branches in the history
- Undo in the Pipeline means going up in History
- No Undo in History: once you delete a branch, it's gone for good!

# Setting up parameters

```

[
  {'J': '1', 'J1': 0.0},
  {'J': '1', 'J1': 0.11},
  ...
]
    
```

# DataSet modules

- DataSet synopsis:

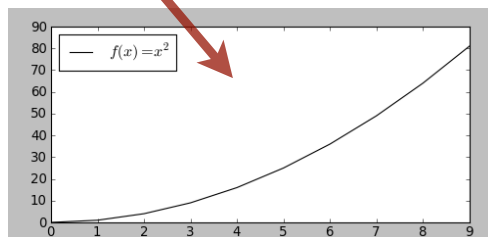
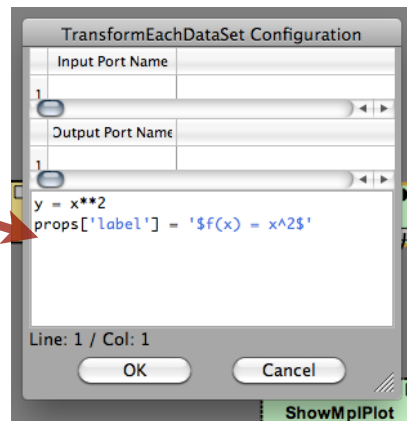
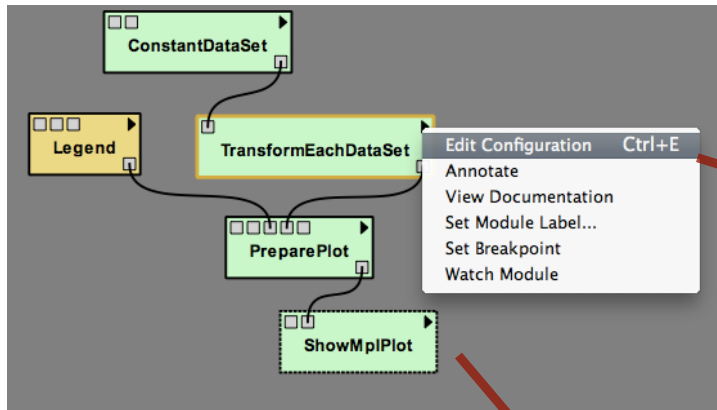
```
class DataSet:
    def __init__(self):
        self.x = np.array([])
        self.y = np.array([])
        self.props = {}
```

- `x/y` should contain data, `props` should be a dictionary containing all information about the data
- DataSet modules exchange lists of DataSets
- Most modules will imply loop over all entries in the list

# More on DataSets

- Transform modules allow arbitrary operations
- Plotting tools will read information (labels, line colors/type, ...) from props, including
  - `'label', 'line', 'linewidth', 'xlabel'/'ylabel', 'marker', 'markersize'`
- Error bars will be shown if data type has `mean/error` members
- Such data types that support error propagation in calculations are part of `pyalps.pyalea`

# A simple workflow



## Variables:

- x
- y
- props

# Hierarchical datasets

- Problem: find minimal energy for each  $t$  in the list

```
[(x=[],y=[],props={t:1,V:1}),
 (x=[],y=[],props={t:1,V:2}),
 (x=[],y=[],props={t:2,v:1}),
 (x=[],y=[],props={t:2,V:2})]
```

- In a flat list: cumbersome

```
mins = {}
for ds in data:
    if ds.props[t] in mins:
        mins[ds.props[t]] = min(min(ds.y), mins[ds.props[t]])
    else:
        mins[ds.props[t]] = min(ds.y)
```

# Hierarchical datasets

- Consider hierarchical list

```
[[ (x=[],y=[],props={t:1,v:1}),
  (x=[],y=[],props={t:1,v:2}),
  (x=[],y=[],props={t:2,v:1}),
  (x=[],y=[],props={t:2,v:2}) ]]
```

- Much easier now:

```
mins = {}
for ds in data:
    mins[ds[0].props[t]] = min([min(v.y) for v in ds])
```

- Functions for grouping/flattening, applying transformations at any level etc. exist

# The spreadsheet

Method	Signature
SheetReference	
MinColumnCount	(Integer)
MinRowCount	(Integer)
SheetName	(String)
Module	

Set Methods

SheetName

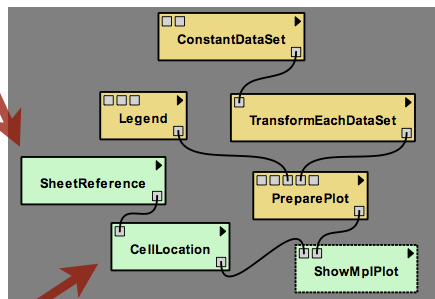
String New sheet

MinColumnCount

Integer 1

MinRowCount

Integer 1



Method	Signature
CellLocation	
Column	(Integer)
ColumnRowAddress	(String)
ColumnSpan	(Integer)
Row	(Integer)
RowSpan	(Integer)

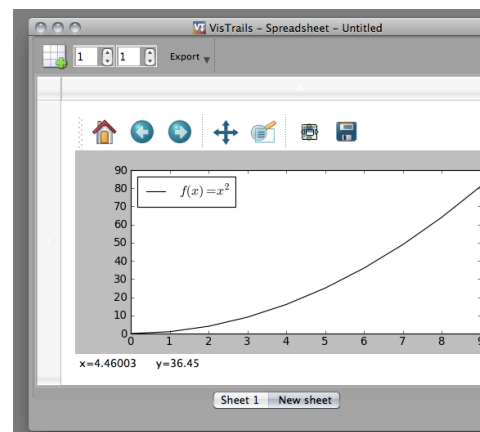
Set Methods

Column

Integer 1

Row

Integer 1



# Exploration

The screenshot displays the VisTrails interface for a workflow named 'fit\_example.vt'. On the left, a workflow diagram shows modules: 'ConstantDataSet', 'TransformEachDataSet', 'RestrictXRange', 'Legend', 'Prepare', and 'ShowMpl'. The central workspace contains three plots of the function  $f(x) = 1 + x$  with different fitted parameters  $A_0$ : 0.69314718056, 0.560947584526, and 0.479246011546. The right-hand panel shows the 'Parameters' section with 'RestrictXRange :: max x' set to 3, and the 'Set Methods' section listing 'ConstantDataSet', 'DoNonlinearFit', 'RestrictXRange', and 'TransformEachDataSet'. Below this is an 'Annotated Pipeline' diagram and a 'Spreadsheet Virtual Cell' section.

# Odds and ends

- How to get more error messages on Mac? Try:  
`vispython /Applications/VisTrails/vistrails.app/Contents/Resources/vistrails.py`
- Screencast introduction to VT and some DataSet modules: <http://alps.comp-phys.org/static/screencast>